Titanic: Machine Learning from Disaster

10446025 張凱勛
10446028 秦仲廷
10346018 鄭彥威

Competition Description

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

Data Introduction

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Variable Notes

pclass: A proxy for socio-economic status (SES)
1st = Upper
2nd = Middle
3rd = Lower

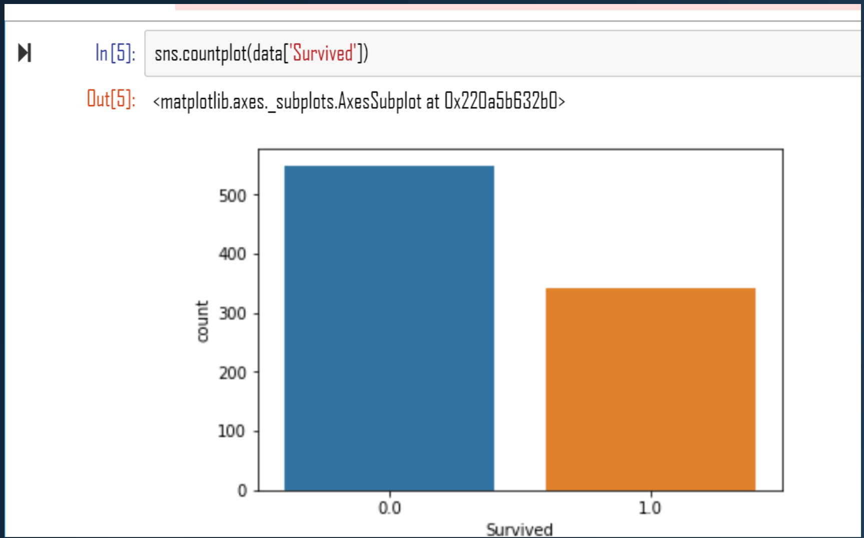
age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...
Sibling = brother, sister, stepbrother, stepsister
Spouse = husband, wife (mistresses and fiancés were ignored)

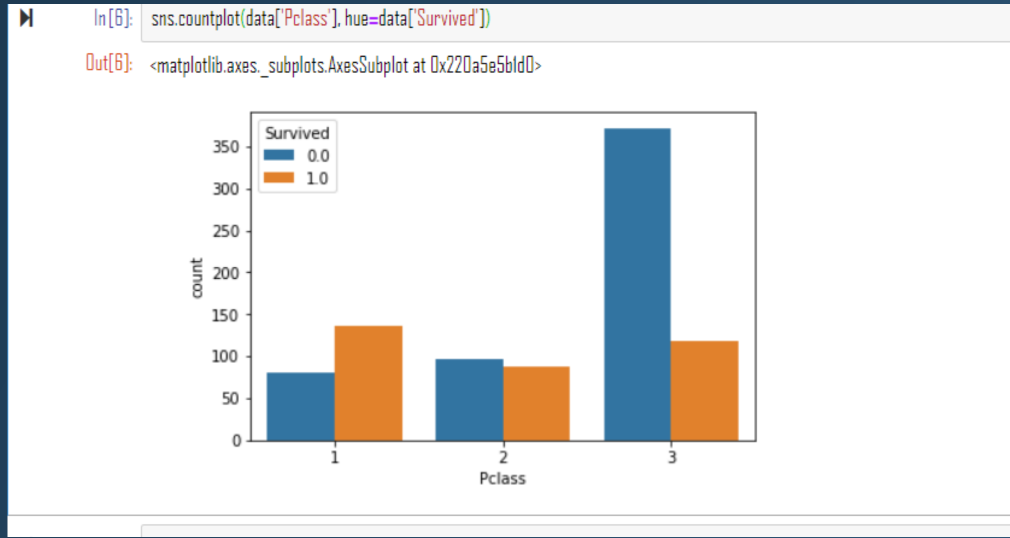
parch: The dataset defines family relations in this way...
Parent = mother, father
Child = daughter, son, stepdaughter, stepson
Some children travelled only with a nanny, therefore parch=0 for them.

Data Analysis

接下來要對資料開始做一些觀察以及分析。首先分析生存以及死亡的比例是否有相當大的落差，發現大概死亡的比例是6成、生存的比例大概是4成



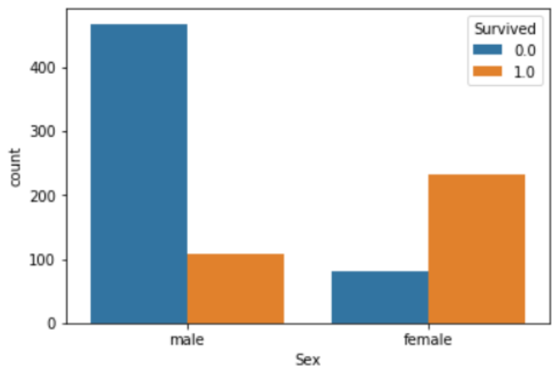
觀察艙等跟生存率的關係，可以發現在1艙等的生存率最高、再來是2艙等、最後是3艙等的



Data Analysis

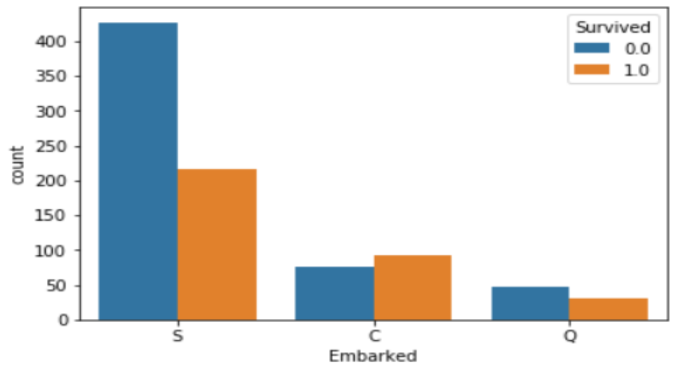
再來是觀察性別跟生存率的關係，發現女生生存率是男生的好幾倍。或許是像在電影裡頭一樣，在逃難的時候先讓女生以及小孩先搭船

```
In [7]: sns.countplot(data['Sex'], hue=data['Survived'])
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x220a5ec7780>
```



出發港口跟生存率的差異，可以發現S港出發的都比較容易死亡，其原因可能是S城市出發的人買的票價都比較便宜

```
In [8]: sns.countplot(data['Embarked'], hue=data['Survived'])
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x220a5f1b6a0>
```



Deep Learning Approach

```
In [19]:
dataAgeNull = data[data["Age"].isnull()]
dataAgeNotNull = data[data["Age"].notnull()]
remove_outlier = dataAgeNotNull[(np.abs(dataAgeNotNull["Fare"]-dataAgeNotNull["Fare"].mean())>(4*dataAgeNotNull["Fare"].std()))]
rfModel_age = RandomForestRegressor(n_estimators=2000,random_state=42)
ageColumns = ['Embarked', 'Fare', 'Pclass', 'Sex','Cabin']
rfModel_age =  rfModel_age.fit(remove_outlier[ageColumns], remove_outlier["Age"])
dataAgeNotNull[ageColumns].head()
```

Out[19]:

	Embarked	Fare	Pclass	Sex	Cabin
0	2	7.2500	2	1	7
1	0	71.2833	0	0	2
2	2	7.9250	2	0	7
3	2	53.1000	0	0	2
4	2	8.0500	2	1	7

```
In [20]:
ageNullValues = rfModel_age.predict(X= dataAgeNull[ageColumns])
```

```
In [21]:
dataAgeNull.loc[:,"Age"] = ageNullValues
data = dataAgeNull.append(dataAgeNotNull)
data.reset_index(inplace=True, drop=True)
```

使用隨機森林來推測年齡

```
In [22]:
dataTrain = data[pd.notnull(data['Survived'])].sort_values(by=["PassengerId"])
dataTest = data[~pd.notnull(data['Survived'])].sort_values(by=["PassengerId"])
```

```
In [23]:
dataTrain = dataTrain[['Survived', 'Age', 'Embarked', 'Fare',  'Pclass', 'Sex' ,'Cabin']]
dataTest = dataTest[['Age', 'Embarked', 'Fare', 'Pclass', 'Sex','Cabin']]
```

```
In [24]:
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(criterion='gini',
                            n_estimators=1000,
                            min_samples_split=12,
                            min_samples_leaf=1,
                            oob_score=True,
                            random_state=1,
                            n_jobs=-1)

rf.fit(dataTrain.iloc[:, 1:], dataTrain.iloc[:, 0])
print("%.4f" % rf.oob_score_)
```

```
0.8204
```

使用隨機森林來預測存活率

Result&Discussion

Passen gerId	Surviv ed	
0	892	0
1	893	0
2	894	0
3	895	0
4	896	1
5	897	0
6	898	0
7	899	0
8	900	1
9	901	0
10	902	0
11	903	0
12	904	1
13	905	0
14	906	1
15	907	1
16	908	0
17	909	0

...	...	...
388	1280	0
389	1281	0
390	1282	1
391	1283	1
392	1284	0
393	1285	0
394	1286	0
395	1287	1
396	1288	0
397	1289	1
398	1290	0
399	1291	0
400	1292	1
401	1293	0
402	1294	1
403	1295	0
404	1296	0
405	1297	1
406	1298	0

最後存活乘客

In [22]:
```
dataTrain = data[pd.notnull(data['Survived'])].sort_values(by=["PassengerId"])
dataTest = data[~pd.notnull(data['Survived'])].sort_values(by=["PassengerId"])
```

In [23]:
```
dataTrain = dataTrain[['Survived', 'Age', 'Embarked', 'Fare',  'Pclass', 'Sex' ,'Cabin']]
dataTest = dataTest[['Age', 'Embarked', 'Fare', 'Pclass', 'Sex','Cabin']]
```

In [24]:
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(criterion='gini',
                n_estimators=1000,
                min_samples_split=12,
                min_samples_leaf=1,
                oob_score=True,
                random_state=1,
                n_jobs=-1)

rf.fit(dataTrain.iloc[:, 1:], dataTrain.iloc[:, 0])
print("%.4f" % rf.oob_score_)
```

0.8204

Entered | Sort by Grouped

All Categories

1 Active Competition

Titanic: Machine Learning from Disaster
Start here! Predict survival on the Titanic and get familiar with ML basics
Getting Started · Ongoing · tutorial, tabular data, binary classification

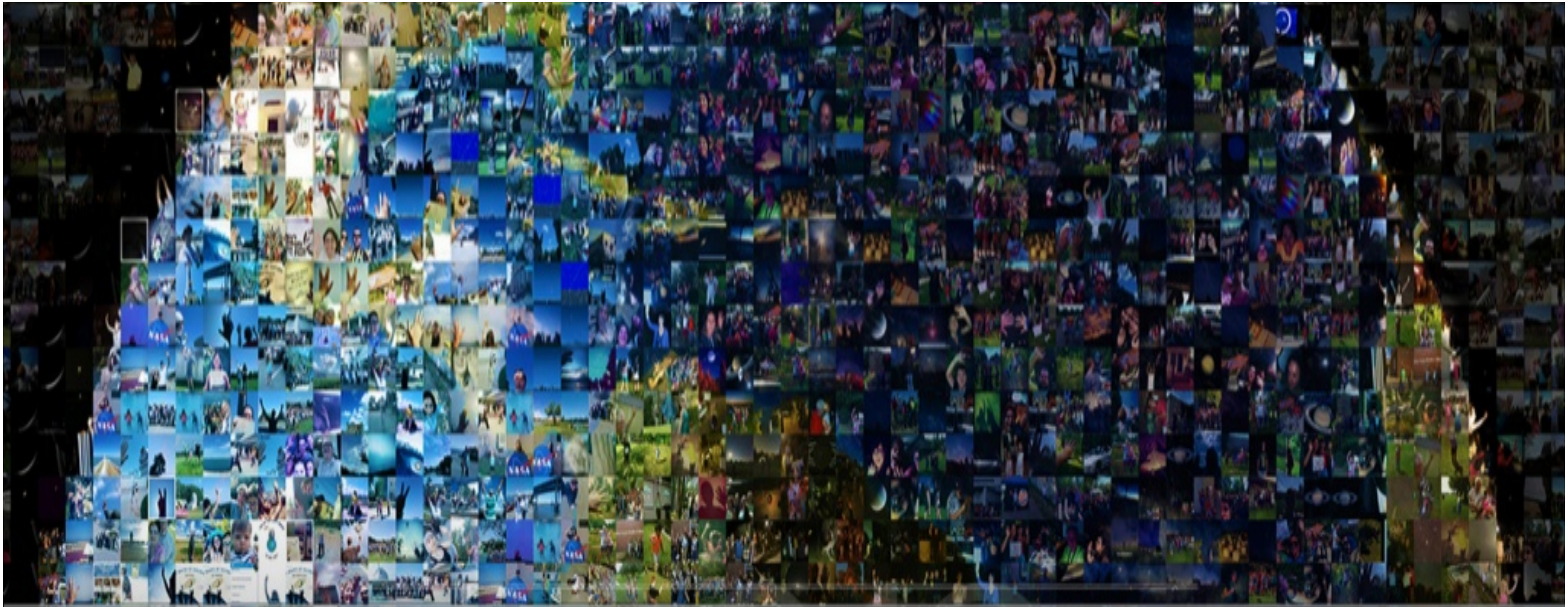
1698/10422
Top 17%

No more competitions to show

加入模型、訓練、觀察oob score
得到了0.8204的oob score， 也有可能是overfittng!，將結果提交至Kaggle

Refrence

- ☑ https://medium.com/@yulon
gtsai
- ☑ https://www.kaggle.com/c/titanic



謝謝觀賞